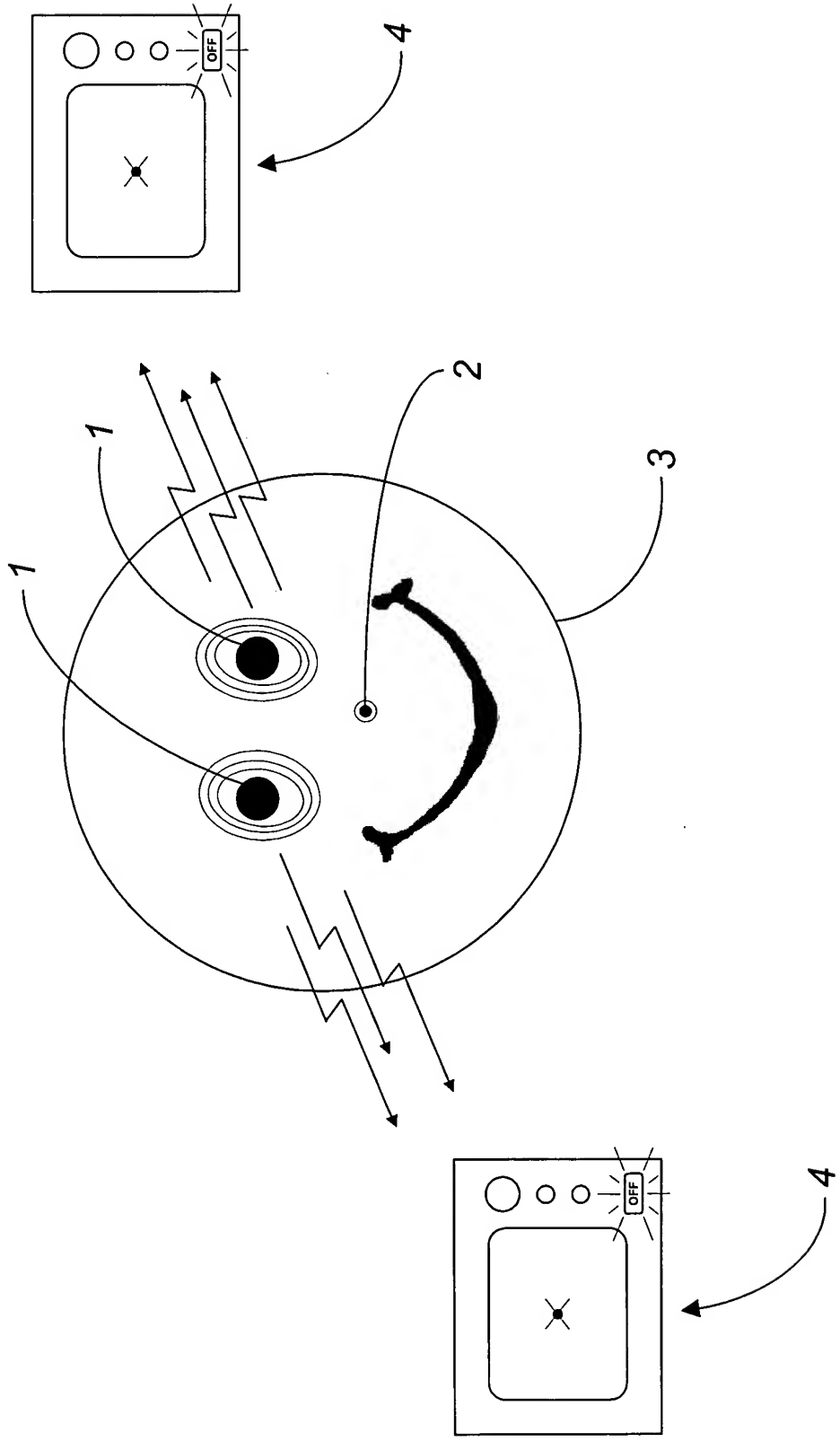

FIG. 1

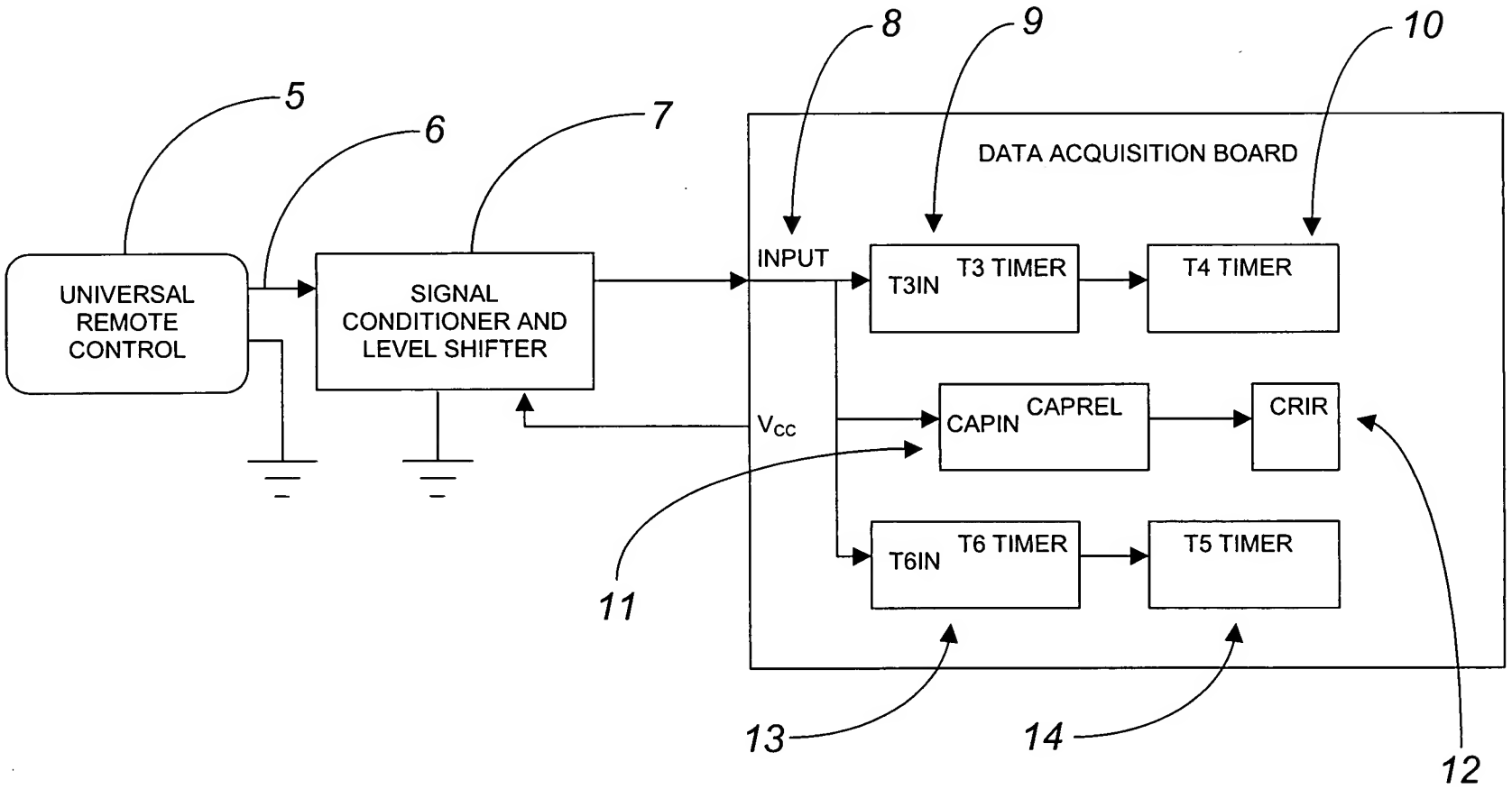
5
6
7
8
9
10
UNIVERSAL REMOTE CONTROL
SIGNAL CONDITIONER AND LEVEL SHIFTER
DATA ACQUISITION BOARD
INPUT
T3IN
T3 TIMER
T4 TIMER
Vcc
CAPIN
CAPREL
CRIR
T6IN
T6 TIMER
T5 TIMER
11
12
13
14

FIG. 2

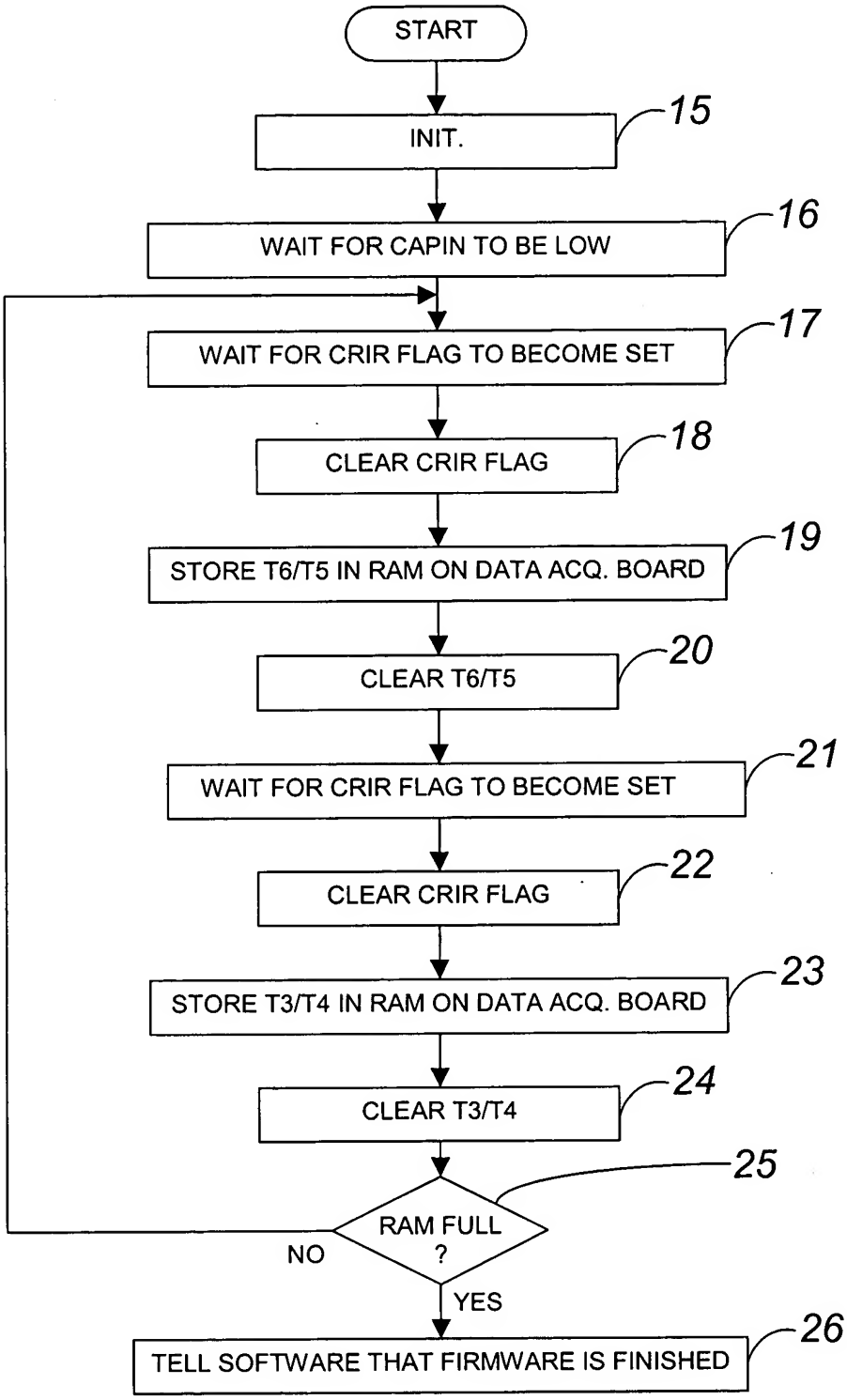
START
INIT.
15
WAIT FOR CAPIN TO BE LOW
16
WAIT FOR CRIR FLAG TO BECOME SET
17
CLEAR CRIR FLAG
18
STORE T6/T5 IN RAM ON DATA ACQ. BOARD
19
CLEAR T6/T5
20
WAIT FOR CRIR FLAG TO BECOME SET
21
CLEAR CRIR FLAG
22
STORE T3/T4 IN RAM ON DATA ACQ. BOARD
23
CLEAR T3/T4
24
RAM FULL ?
25
NO
YES
TELL SOFTWARE THAT FIRMWARE IS FINISHED
26

FIG. 3

27

Carrier Average Low period = 18980 nanosec = 76 counts

Carrier Average High period = 20030 nanosec = 80 counts

On-Times (nanosec)	Off-Times (nanosec)	On-Time (counts)	Off-Time (counts)
1157536	987745	579	494
1162496	982785	581	491
1167296	1991905	584	996
1155296	27044384	578	13522
1161856	983265	581	492
1156576	988385	578	494
1166016	1993345	583	997
1150016	27044384	575	13522

FIG. 4

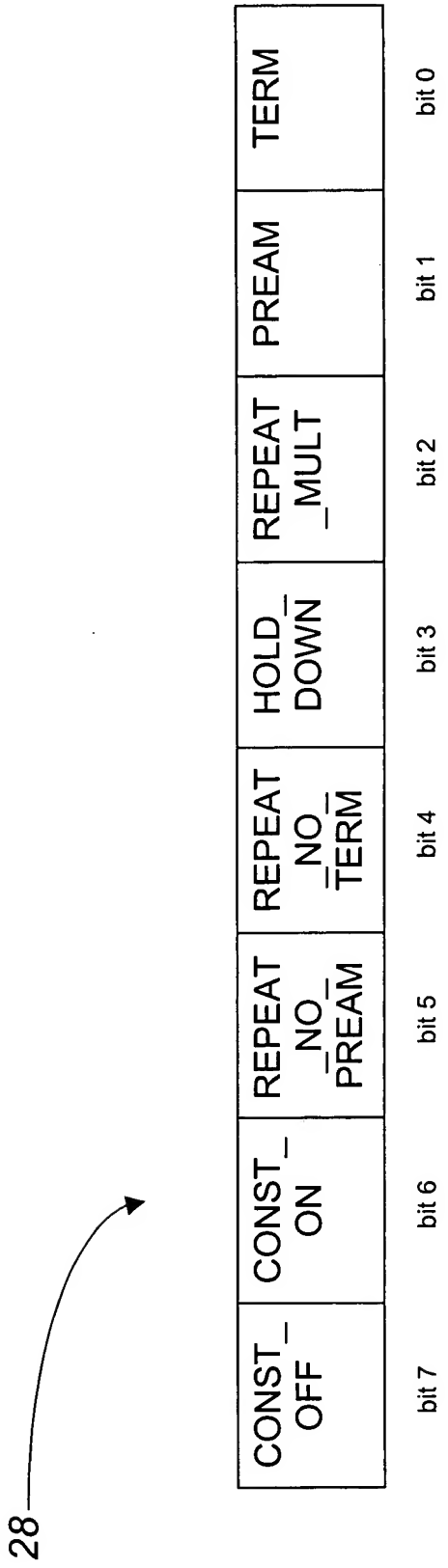
28
CONST_ OFF
CONST_ ON
REPEAT _NO_ PREAM
REPEAT _NO_ TERM
HOLD_ DOWN
REPEAT _MULT
PREAM
TERM
bit 7
bit 6
bit 5
bit 4
bit 3
bit 2
bit 1
bit 0

FIG. 5

29

xx	code_type
xx	Carrier Timer High counts (in counts for Carrier Timer)
xx	Carrier Timer Low counts (in counts for Carrier Timer)
xxxx	Preamble On-Time (in counts for Gating Timer)
xxxx	Preamble Off-Time (in counts for Gating Timer)
xxxx	Const-Time
xx	# Time-Tab entries
xxxxxxxx	time_tab (for CONST_ON or CONST_OFF code_type:
...	2 bytes for each Off-Time or each On-Time
...	for the other two code_type's:
...	4 bytes for each On-Time/Off-Time pair)
xxxxxxxx	(in counts for Gating Timer)
xx	# Sequence Nybbles
x..x	Sequence of Nybbles (pad with a 0 nybble to make an even # of nybbles, if necessary)
xxxx	Terminator On-Time (in counts for Gating Timer)
xxxx	Terminator Off-Time (in counts for Gating Timer)
xx	# Hold-Down Sequence On-Time/Off-Time pairs
xxxxxxxx	Hold-Down Sequence On-Time/Off-Time pairs (in counts for Gating Timer)
...	
xxxxxxxx	

FIG. 6

30

Avg. Low period (counts) = 1E

Avg. High period (counts) = 26

On-Time (counts)	Off-Time (counts)	Sequence offset
699	69E	Preamble
19D	35E	0
19D	35E	0
19D	35E	0
19D	35E	0
19D	35E	0
19D	35E	0
19D	35E	0
19D	35E	0
19D	35E	0
19D	35E	0
19D	696	1
19E	696	1
19E	696	1
19E	696	1
19E	696	1
19E	696	1
19E	696	1
19D	696	1
19D	697	1
19C	697	1
1A1	69C	1
198	363	0
198	3B81	Terminator

FIG. 7

31

Avg. Low period (counts) = 42

Avg. High period (counts) = 22

On-Time (counts)	Off-Time (counts)	Sequence offset
1162	8C0	Preamble
11D	361	1
11E	360	1
120	36B	1
11F	35F	1
120	36B	1
11D	361	1
11E	360	1
120	36B	1
11D	109	0
120	113	0
120	113	0
11E	108	0
11F	113	0
120	113	0
11E	108	0
120	113	0
11F	35F	1
11F	36B	1
11F	35F	1
11C	362	1
129	362	1
11F	35F	1
11F	113	0
120	113	0
11D	109	0
120	113	0
11C	10A	0
129	10A	0
120	113	0
11D	109	0
11F	36B	1
120	35F	1
120	5110	2
1162	459	Hold-Down 0
120	B8E4	Hold-Down 1

FIG. 8

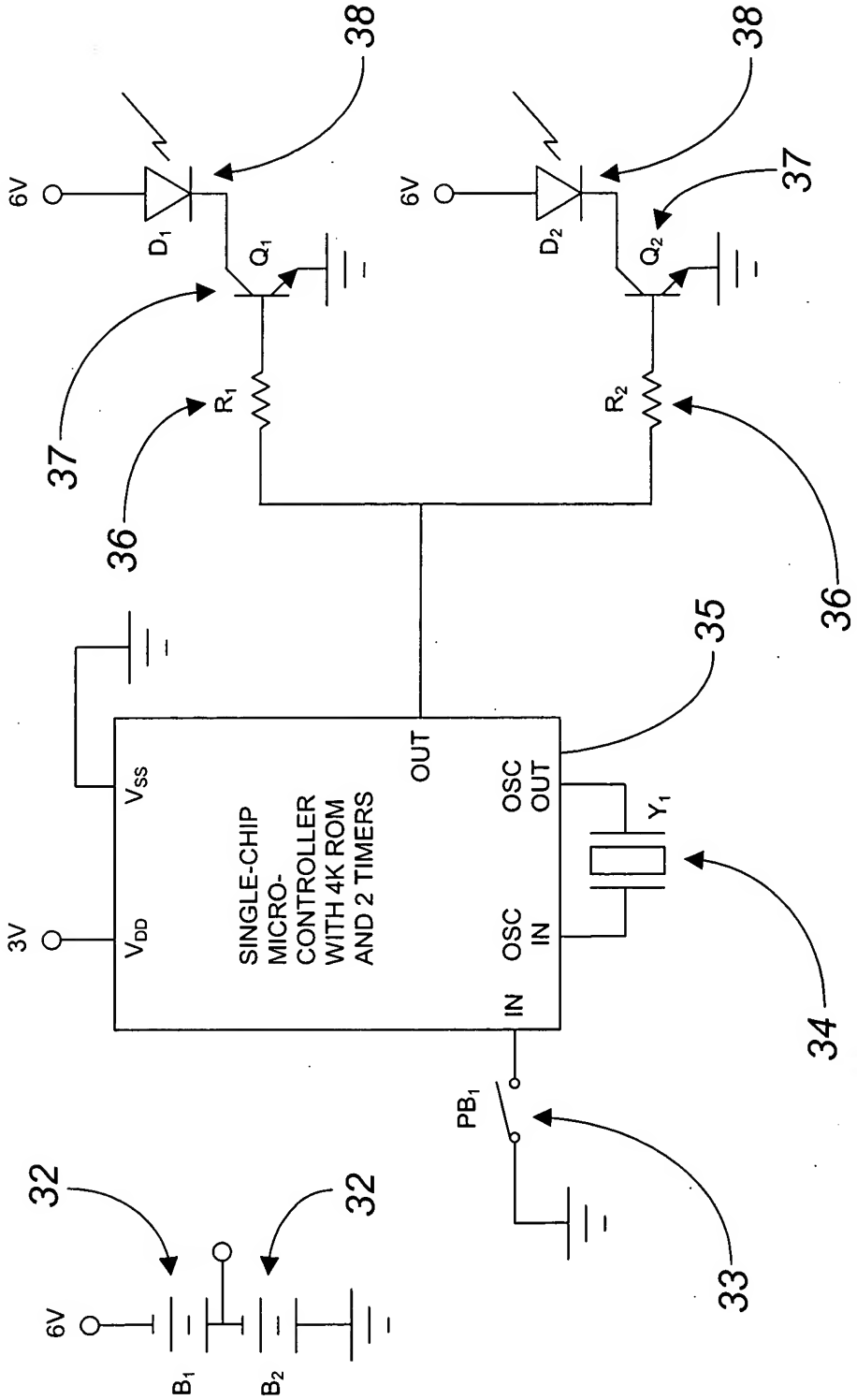
6V
D1
Q1
R1
38
37
36
6V
D2
Q2
R2
38
37
36
35
SINGLE-CHIP
MICRO-
CONTROLLER
WITH 4K ROM
AND 2 TIMERS
VSS
VDD
OUT
OSC
OUT
OSC
IN
IN
3V
Y1
34
PB1
33
32
32
6V
B1
B2

FIG. 9

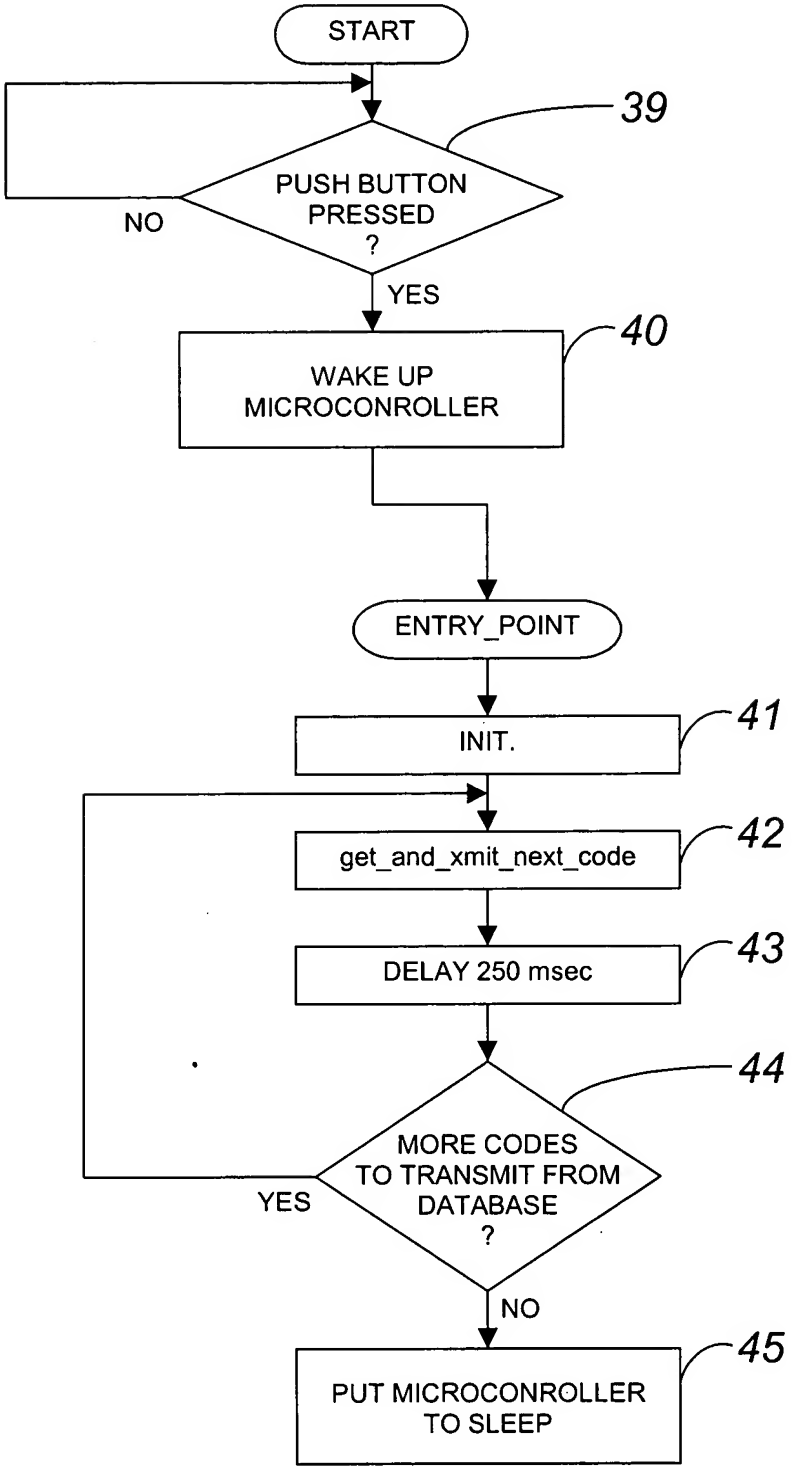
START
39
PUSH BUTTON PRESSED ?
NO
YES
40
WAKE UP MICROCONROLLER
ENTRY_POINT
41
INIT.
42
get_and_xmit_next_code
43
DELAY 250 msec
44
MORE CODES TO TRANSMIT FROM DATABASE ?
YES
NO
45
PUT MICROCONROLLER TO SLEEP

FIG. 10A

FIG. 10B

FIG. $10B_1$
FIG. $10B_2$
FIG. $10B_3$
FIG. $10B_4$
FIG. $10B_5$

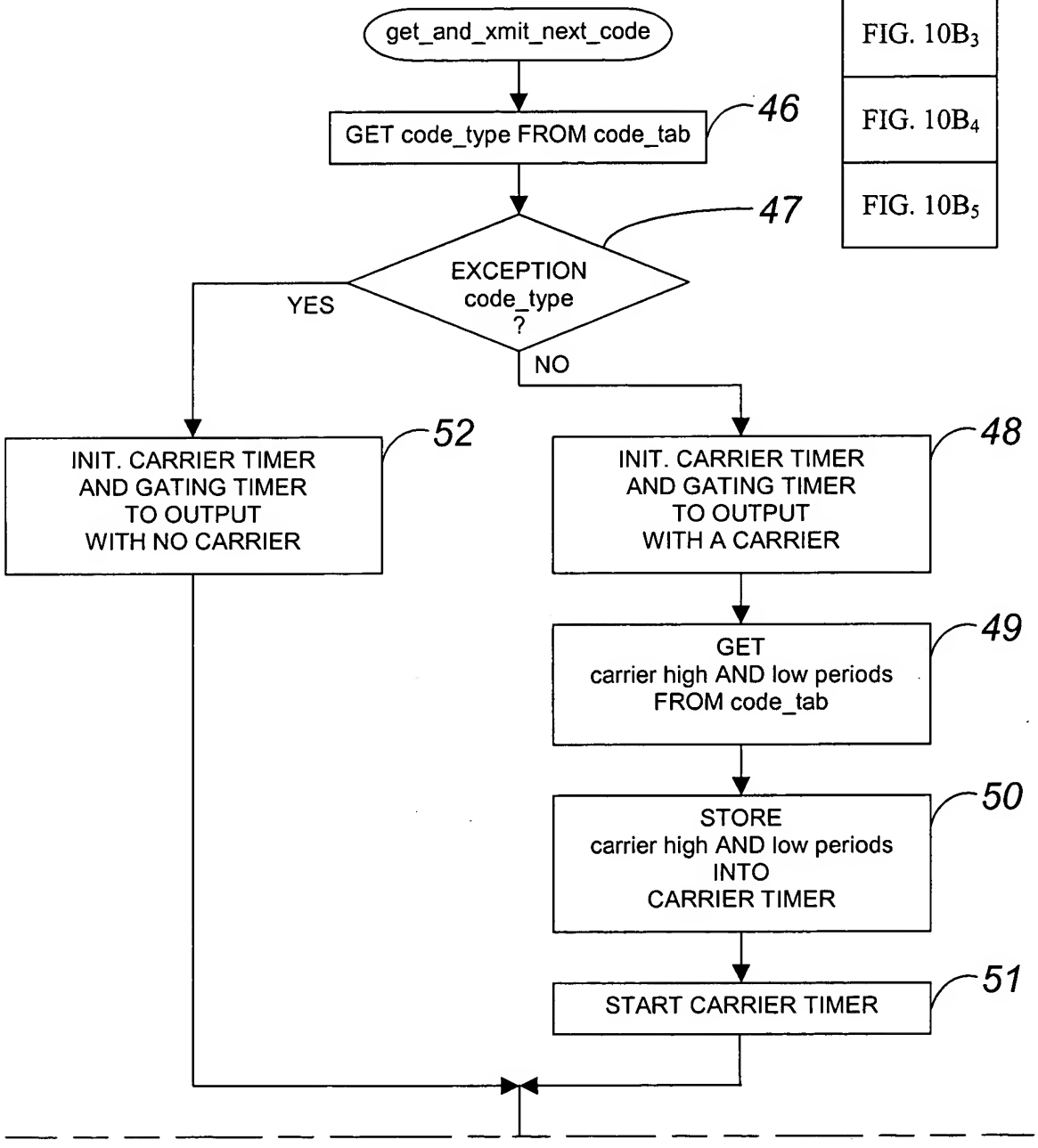

FIG. $10B_1$

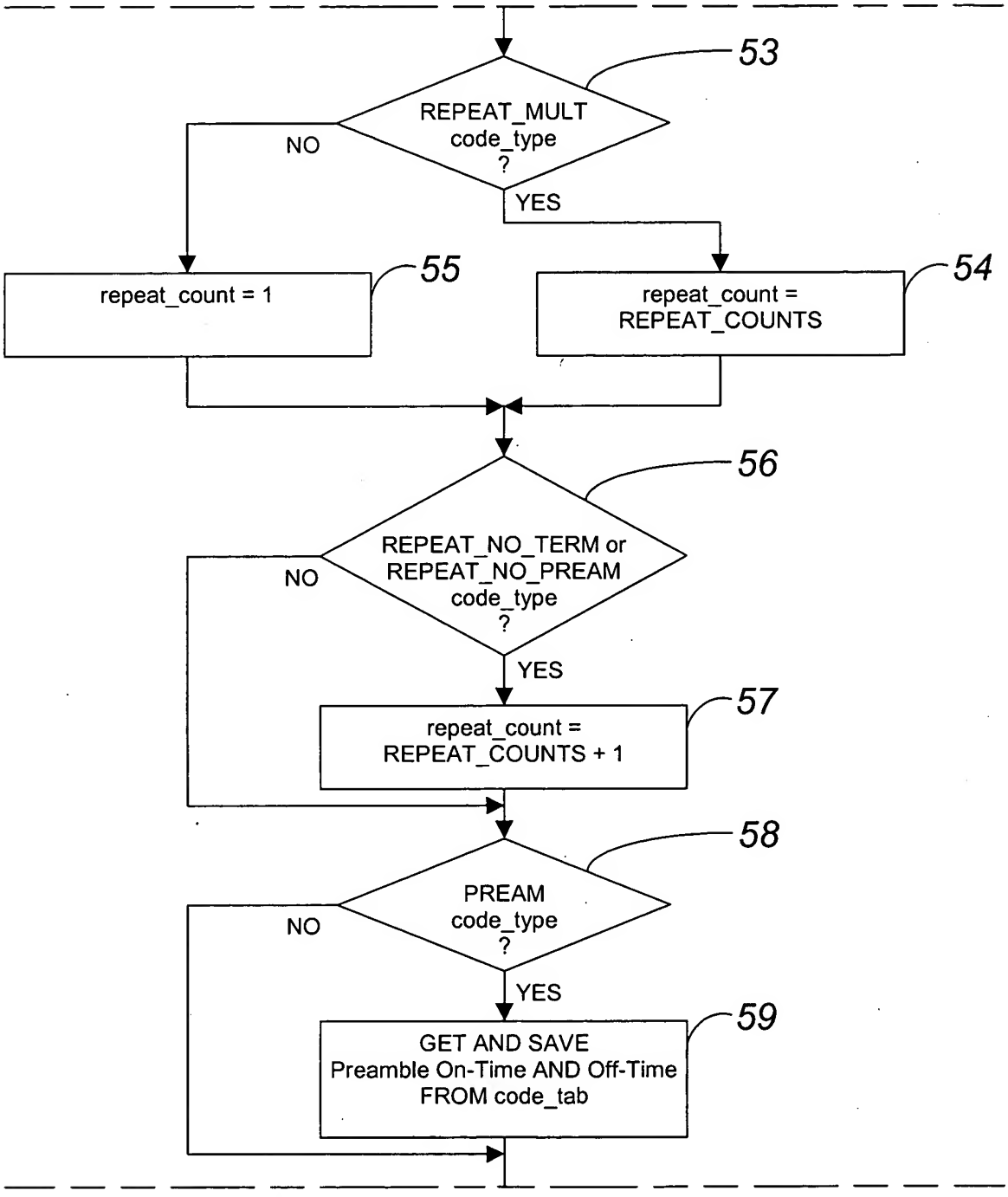
53
REPEAT_MULT
code_type
?
NO
YES
55
repeat_count = 1
54
repeat_count =
REPEAT_COUNTS
56
REPEAT_NO_TERM or
REPEAT_NO_PREAM
code_type
?
NO
YES
57
repeat_count =
REPEAT_COUNTS + 1
58
PREAM
code_type
?
NO
YES
59
GET AND SAVE
Preamble On-Time AND Off-Time
FROM code_tab

FIG. $10B_2$

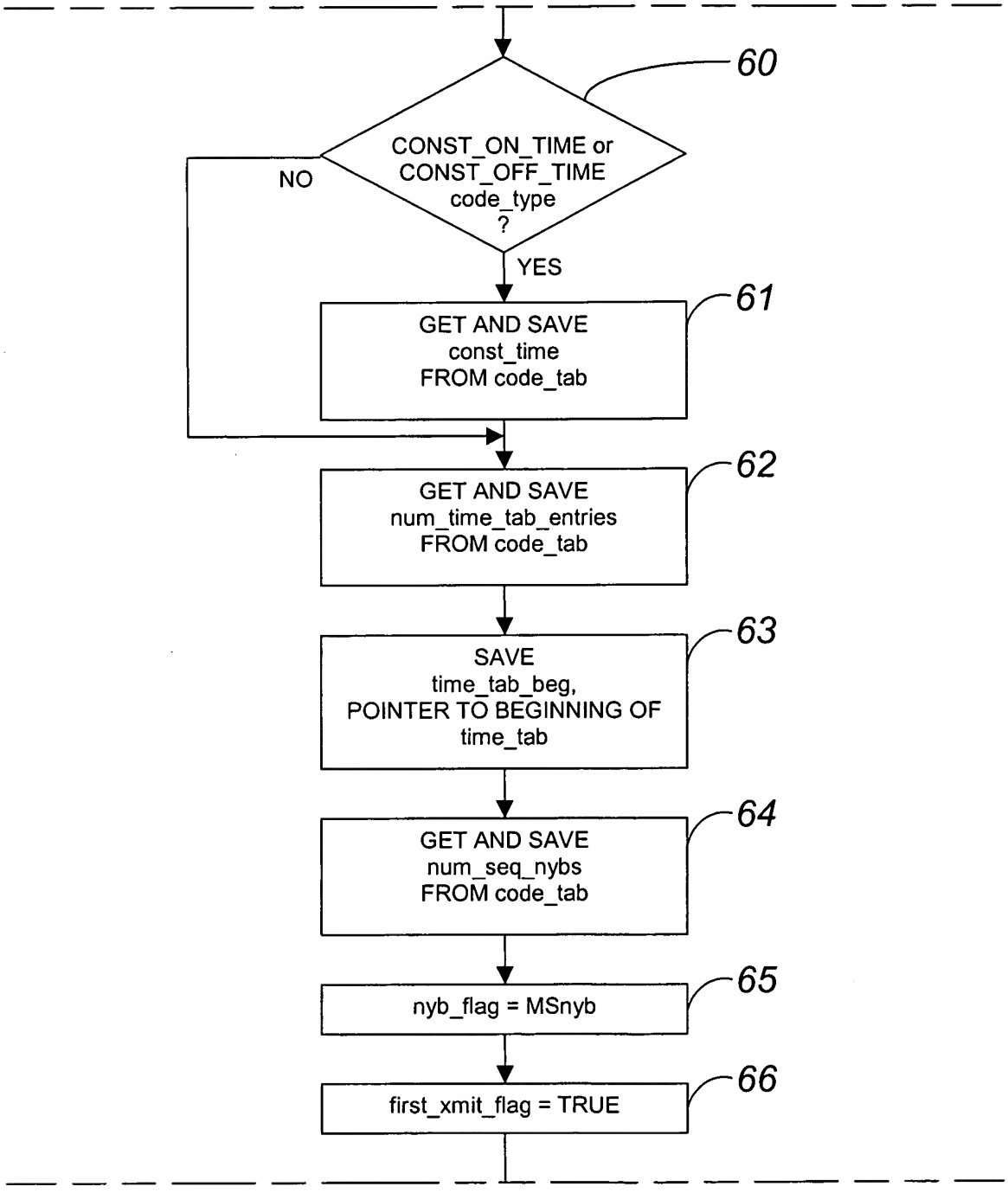
60
CONST_ON_TIME or CONST_OFF_TIME code_type ?
NO
YES
61
GET AND SAVE const_time FROM code_tab
62
GET AND SAVE num_time_tab_entries FROM code_tab
63
SAVE time_tab_beg, POINTER TO BEGINNING OF time_tab
64
GET AND SAVE num_seq_nybs FROM code_tab
65
nyb_flag = MSnyb
66
first_xmit_flag = TRUE

FIG. 10B$_3$

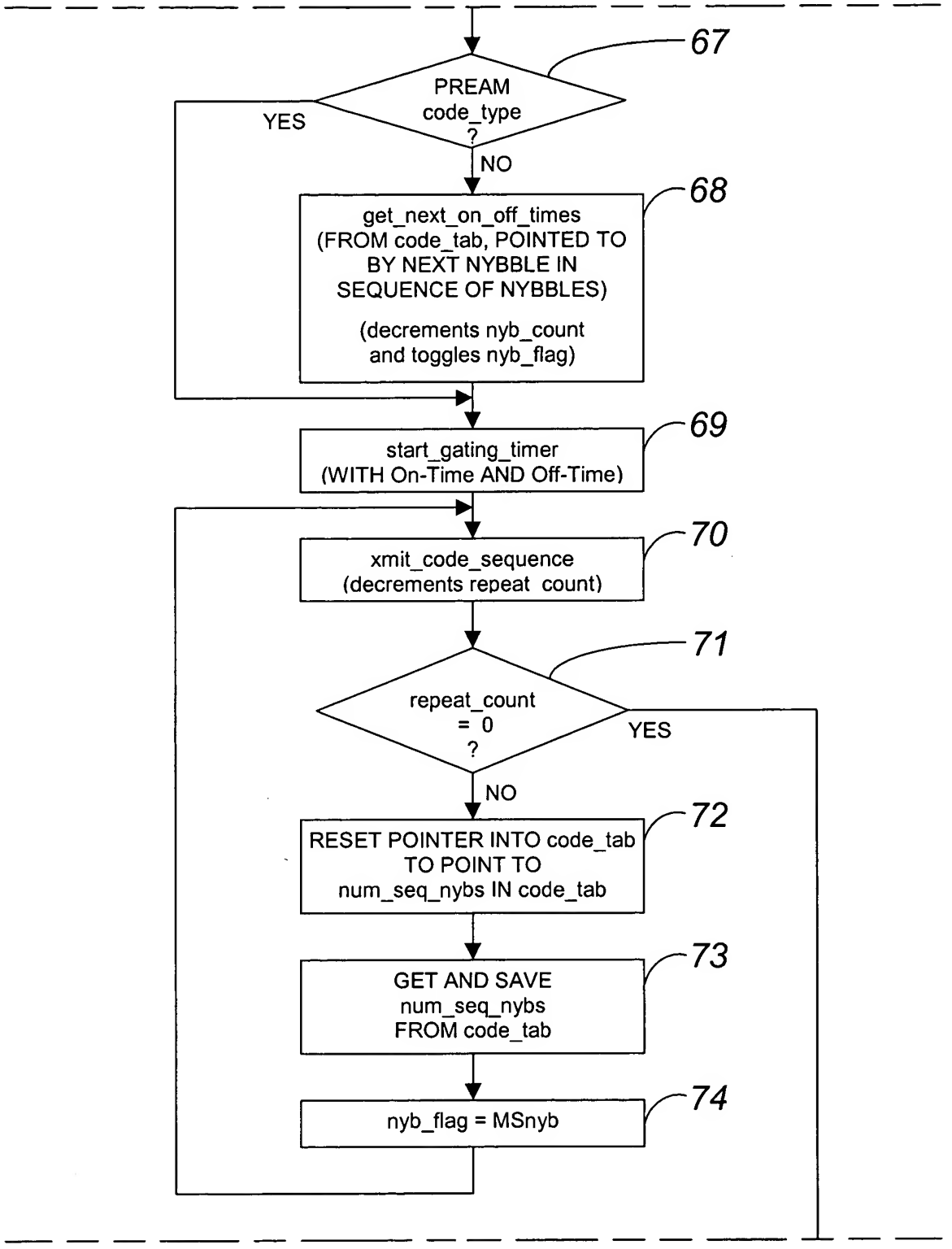
67
PREAM
code_type
?
YES
NO
68
get_next_on_off_times
(FROM code_tab, POINTED TO
BY NEXT NYBBLE IN
SEQUENCE OF NYBBLES)
(decrements nyb_count
and toggles nyb_flag)
69
start_gating_timer
(WITH On-Time AND Off-Time)
70
xmit_code_sequence
(decrements repeat count)
71
repeat_count
= 0
?
YES
NO
72
RESET POINTER INTO code_tab
TO POINT TO
num_seq_nybs IN code_tab
73
GET AND SAVE
num_seq_nybs
FROM code_tab
74
nyb_flag = MSnyb

FIG. 10B$_4$

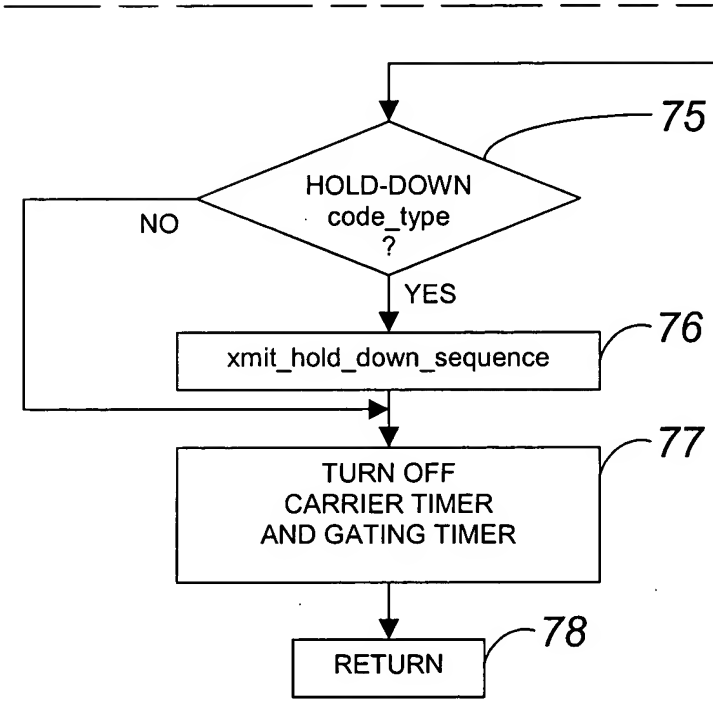
75
HOLD-DOWN
code_type
?
NO
YES
76
xmit_hold_down_sequence
77
TURN OFF
CARRIER TIMER
AND GATING TIMER
78
RETURN

FIG. 10B$_5$

FIG. 10C

FIG. $10C_1$
FIG. $10C_2$
FIG. $10C_3$

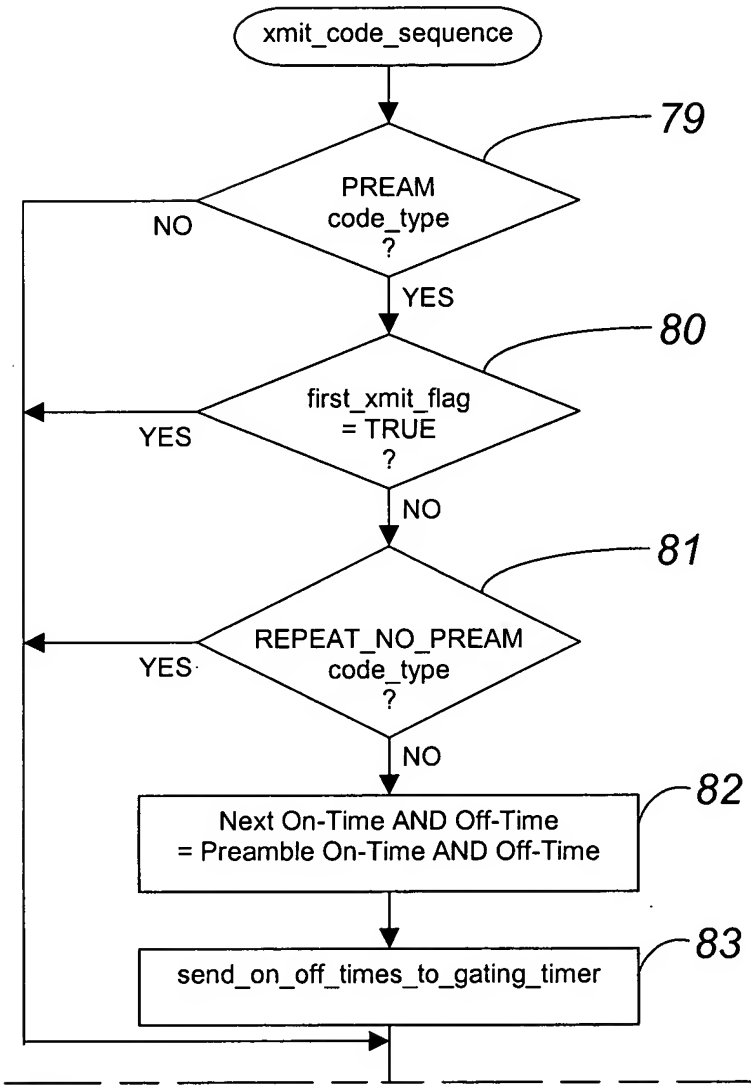

FIG. $10C_1$

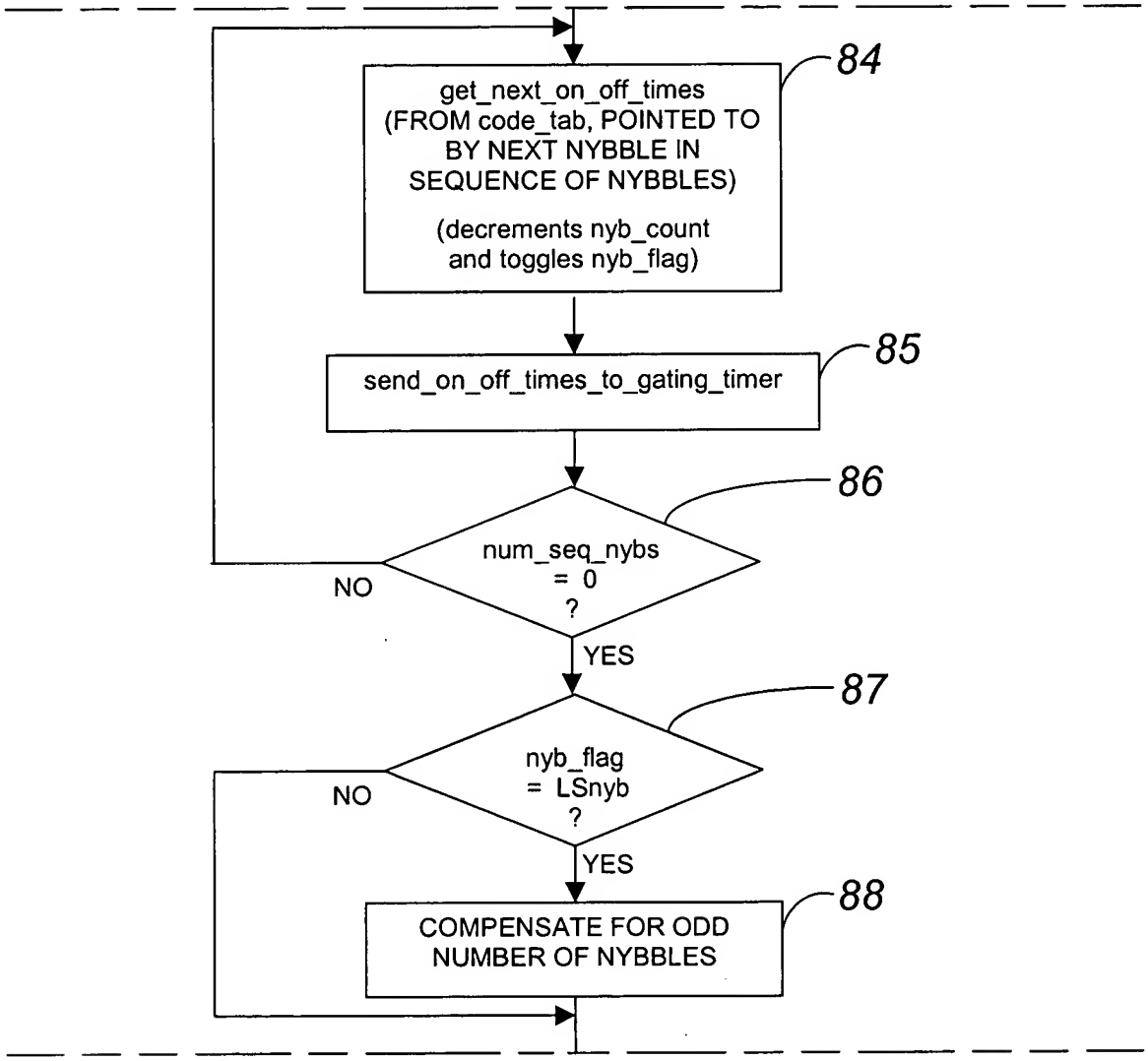
get_next_on_off_times
(FROM code_tab, POINTED TO BY NEXT NYBBLE IN SEQUENCE OF NYBBLES)
(decrements nyb_count and toggles nyb_flag)
84
send_on_off_times_to_gating_timer
85
num_seq_nybs = 0 ?
86
NO
YES
nyb_flag = LSnyb ?
87
NO
YES
COMPENSATE FOR ODD NUMBER OF NYBBLES
88

FIG. 10C$_2$

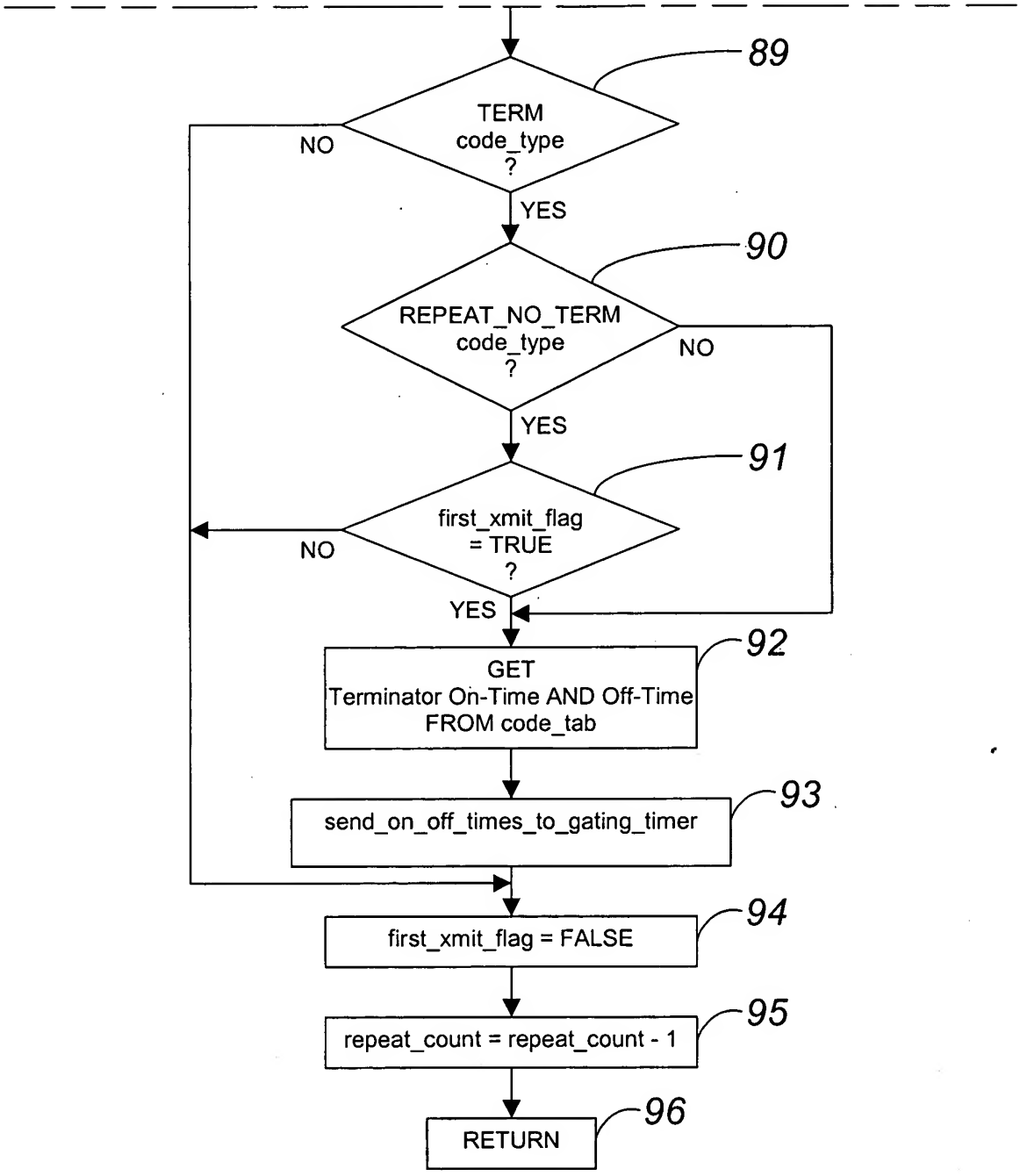
89
TERM
code_type
?
NO
YES
90
REPEAT_NO_TERM
code_type
?
NO
YES
91
first_xmit_flag
= TRUE
?
NO
YES
92
GET
Terminator On-Time AND Off-Time
FROM code_tab
93
send_on_off_times_to_gating_timer
94
first_xmit_flag = FALSE
95
repeat_count = repeat_count - 1
96
RETURN

FIG. 10C$_3$

FIG. 10D

FIG. $10D_1$
FIG. $10D_2$

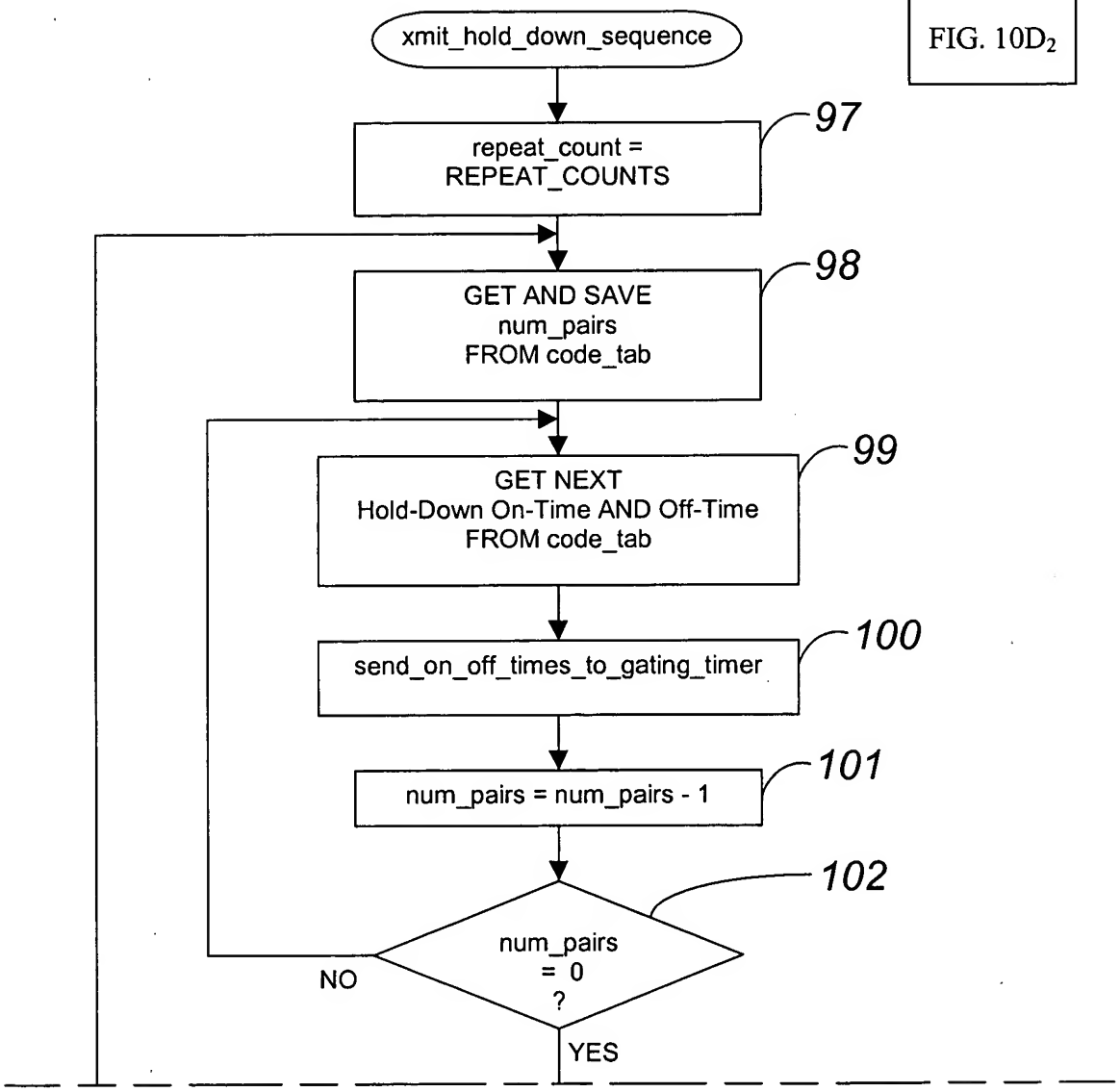

FIG. $10D_1$

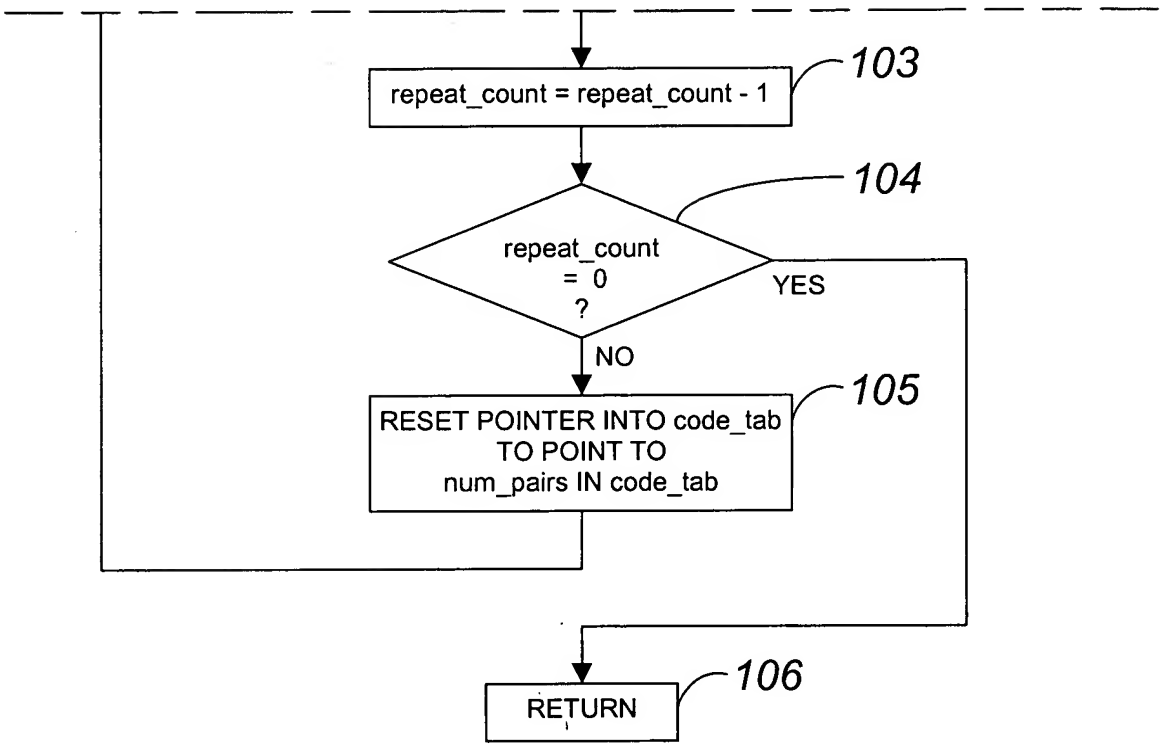
repeat_count = repeat_count - 1
103
repeat_count
= 0
?
104
YES
NO
RESET POINTER INTO code_tab
TO POINT TO
num_pairs IN code_tab
105
RETURN
106

FIG. 10D$_2$

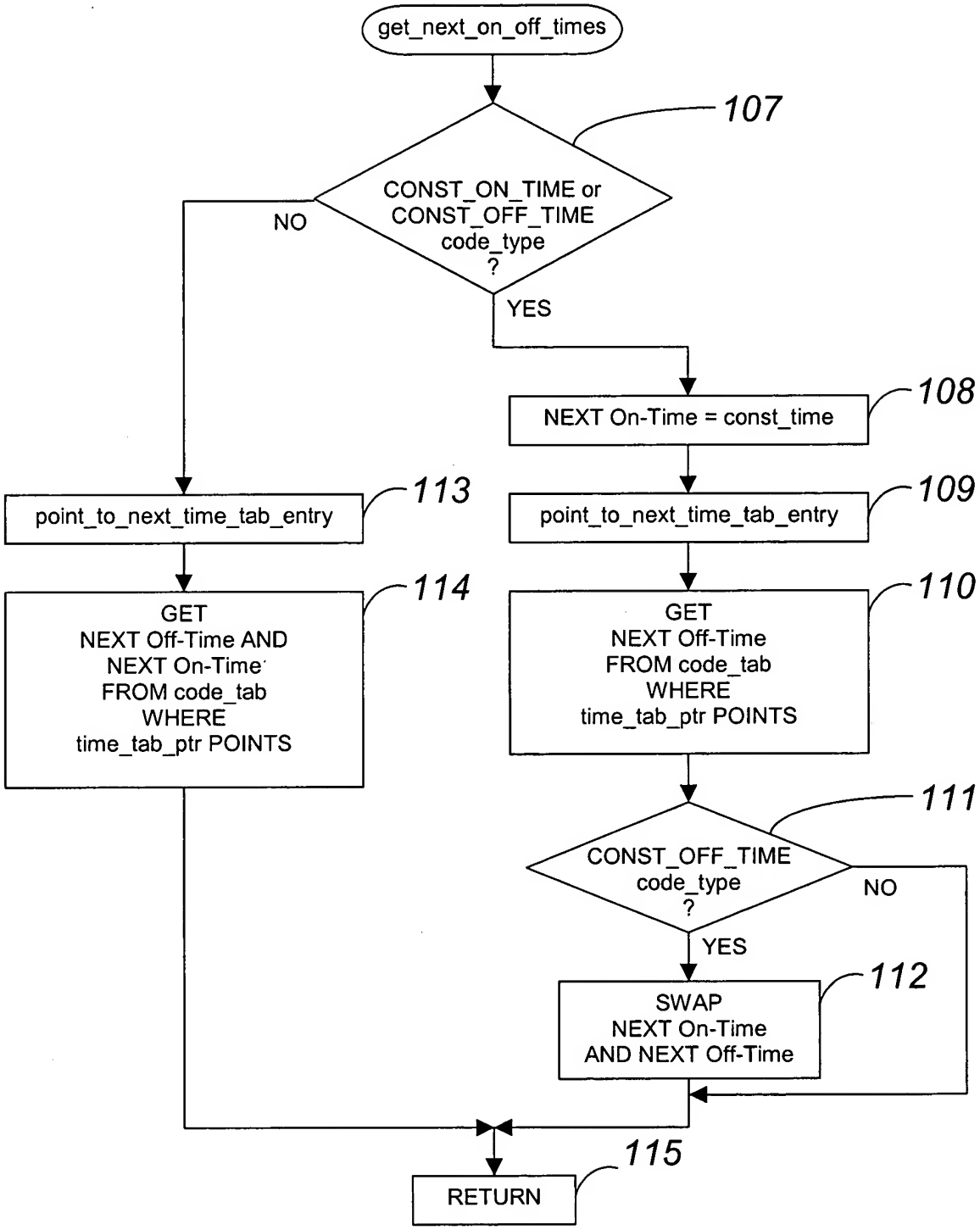
get_next_on_off_times
107
CONST_ON_TIME or
CONST_OFF_TIME
code_type
?
NO
YES
108
NEXT On-Time = const_time
113
point_to_next_time_tab_entry
109
point_to_next_time_tab_entry
114
GET
NEXT Off-Time AND
NEXT On-Time
FROM code_tab
WHERE
time_tab_ptr POINTS
110
GET
NEXT Off-Time
FROM code_tab
WHERE
time_tab_ptr POINTS
111
CONST_OFF_TIME
code_type
?
NO
YES
112
SWAP
NEXT On-Time
AND NEXT Off-Time
115
RETURN

FIG. 10E

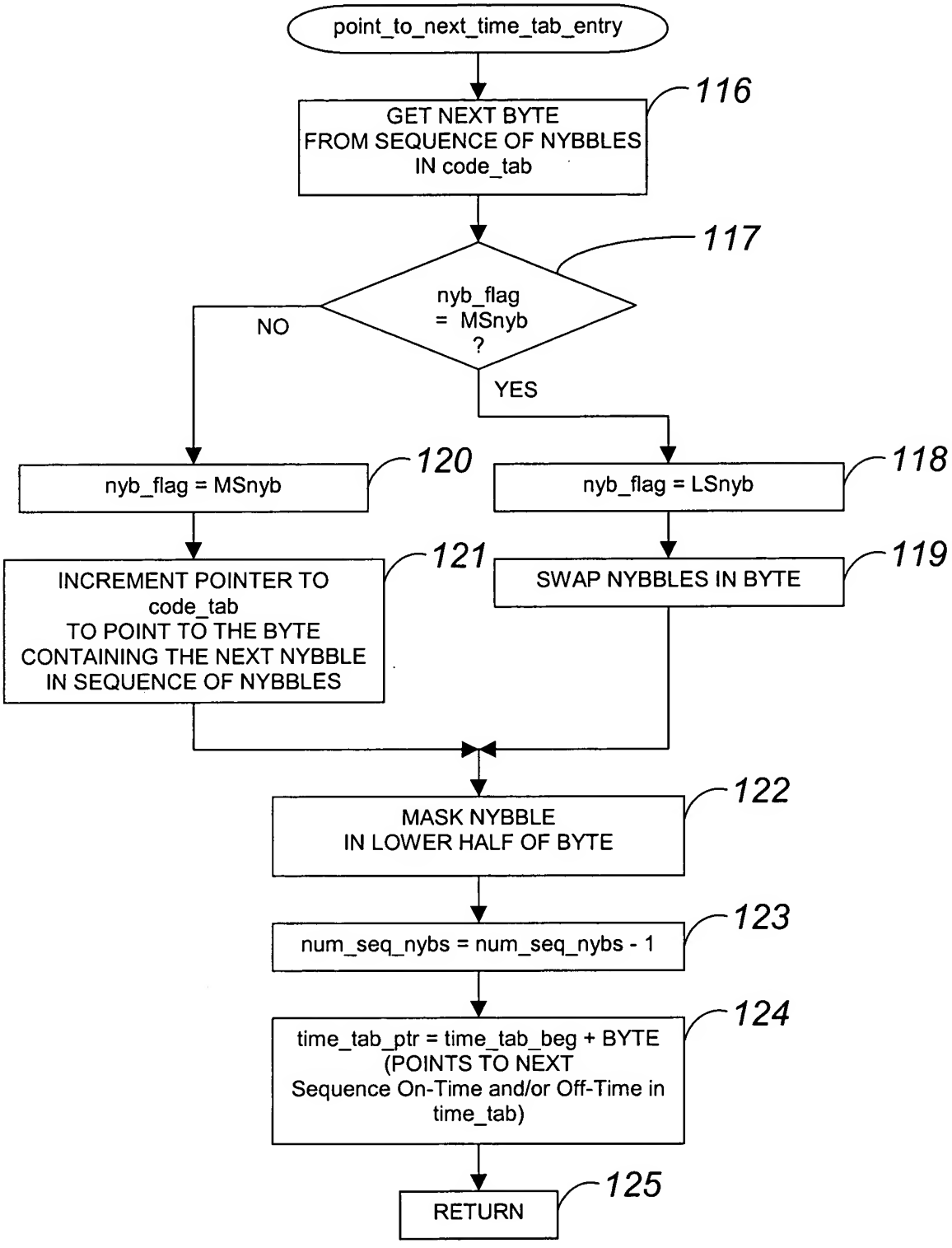
point_to_next_time_tab_entry
GET NEXT BYTE FROM SEQUENCE OF NYBBLES IN code_tab
116
nyb_flag = MSnyb ?
117
NO
YES
nyb_flag = MSnyb
120
nyb_flag = LSnyb
118
INCREMENT POINTER TO code_tab TO POINT TO THE BYTE CONTAINING THE NEXT NYBBLE IN SEQUENCE OF NYBBLES
121
SWAP NYBBLES IN BYTE
119
MASK NYBBLE IN LOWER HALF OF BYTE
122
num_seq_nybs = num_seq_nybs - 1
123
time_tab_ptr = time_tab_beg + BYTE (POINTS TO NEXT Sequence On-Time and/or Off-Time in time_tab)
124
RETURN
125

FIG. 10F

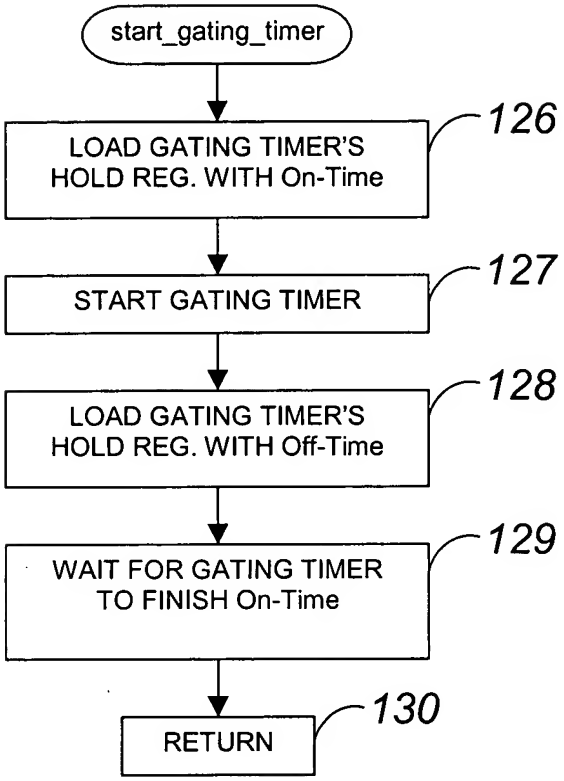
start_gating_timer
LOAD GATING TIMER'S HOLD REG. WITH On-Time
126
START GATING TIMER
127
LOAD GATING TIMER'S HOLD REG. WITH Off-Time
128
WAIT FOR GATING TIMER TO FINISH On-Time
129
RETURN
130

FIG. 10G

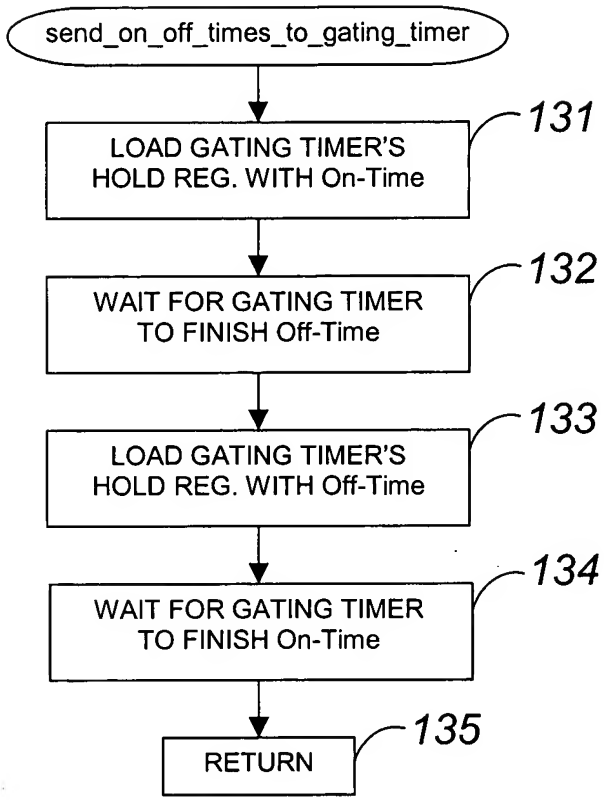
send_on_off_times_to_gating_timer
LOAD GATING TIMER'S HOLD REG. WITH On-Time
131
WAIT FOR GATING TIMER TO FINISH Off-Time
132
LOAD GATING TIMER'S HOLD REG. WITH Off-Time
133
WAIT FOR GATING TIMER TO FINISH On-Time
134
RETURN
135

FIG. 10H